Cocoa Today

Session 102

These are confidential sessions—please refrain from filming or taking pictures.

Application Technologies

Cocoa Today

Matthew Formica
Cocoa & Developer Tools Evangelist

http://developer.apple.com/wwdc2005/

What You'll Learn

- Cocoa: Apple's RAD framework
- Today: A growing developer base…
  - …in a Mac OS X Tiger world,
  - With a Jaguar and Panther history,
  - And a technology foundation that continues to evolve

“We thought we would see how hard it would be to switch the code we had to use Cocoa Bindings. We rewrote everything in a day or two—I think we deleted over a thousand lines of code that just wasn’t needed any more.”

Wil Shipley, Co-founder, Delicious Monster

“Without Cocoa Bindings, it would have taken another four or five months, maybe more, to finish Delicious Library.”

Wil Shipley, Co-founder, Delicious Monster

“With Cocoa, we were able to accomplish a lot more, more quickly.”

Jerry Halls, Vice President, Chronos—maker of StickyBrain

“Be bold—Take the leap.”

Jerry Halls, Vice President, Chronos—Maker of StickyBrain

“I want to marry Core Data and have its children.”

James Dessart, skwirl.net May 3, 2005

What Will We Talk About?

- Design, Architecture, and Tools
- Recent Changes and Directions
- Testimonial

Design, Architecture, and Tools

Ali Ozer
Manager, Cocoa Frameworks

http://developer.apple.com/wwdc2005/

Cocoa Goals

- "Simple Things Simple, Complex Things Possible"
- We write the "common" code for you
  - Behaviors all applications share
- We make it possible for you to do much more
  - Customizable
  - Extensible

Cocoa Is Not a Separate "Box"

- Not isolated in its own environment
- Can easily interact with other subsystems
- Objective-C is a thin superset of C
- Does not provide API for every technology on the system
  - Usually those where Cocoa's object oriented approach provides additional benefits
    - Abstraction
    - Power
    - Rapid development
    - Performance

Performance

- "Framework" or "object-oriented" do not imply "inefficient"
- Cocoa was developed on machines with 4MB of memory and 25MHz processors
- Objective-C is object-oriented and dynamic, but has
  - Low memory overhead
  - Low CPU overhead
- Object-oriented techniques and abstractions often help boost performance where needed

Application Programming Interface (API)

- Consistent naming and calling conventions
  - APIs that are “impedance-matched”
  - Small number of concepts, widely reused

Universal

- Cocoa went through the hoops
  - 68030
  - RISC
  - x86

Universal

- Architecture independent
  - APIs
  - Abstractions
  - Opaque objects
  - Archiving

Objective-C++

- Enabling integration of C++ code with Objective-C
  - A “peaceful co-existence”
- Runtime systems of both languages remain unchanged
  - “Native” ABI, semantics, and performance
  - Subclassing across object models not supported

Cocoa Java

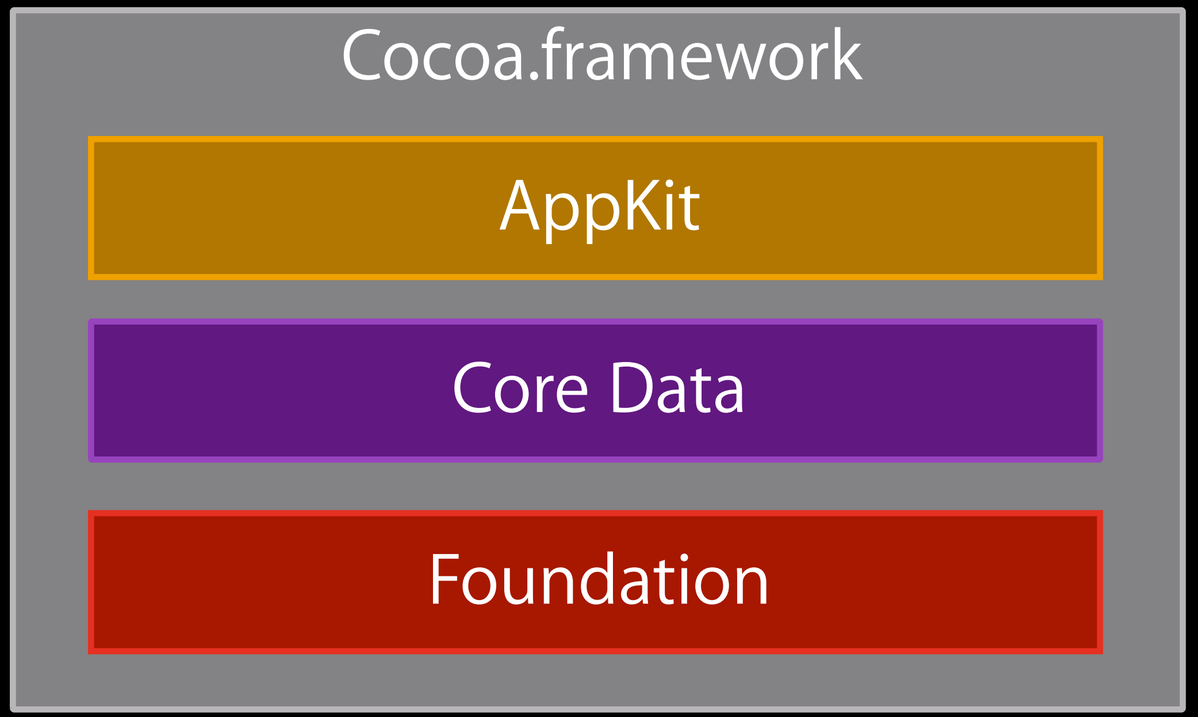
- "Bridging" to make Cocoa APIs available to use from Java
  - For the most part transparent
- However, downsides
  - Setting up and maintaining bridging of new classes isn't automatic
  - Bridging itself isn't "toll-free"
    - Apps have slower startup times and larger footprints than their Objective-C counterparts
- Very few developers are actually using Java to program Cocoa
- We will not make additional Cocoa APIs available in Java

Cocoa/Carbon Interaction

- Supported
  - Non UI code
  - Modal panels in the other environment
- Supported, with caveats
  - Document windows in the other environment
- Not supported
  - Hosting views, sheets, or drawers in windows of the other environment

Core Cocoa Frameworks

- Foundation
- Core Data
- Application Kit (AppKit)



Other Frameworks

- WebKit
- AddressBook
- InstantMessage
- PDFKit
- SyncServices
- QTKit
- CoreImage
- QuartzComposer

Tools

- Xcode
  - Great support for Objective-C and frameworks
  - Class modeling
  - Data modeling
- Interface Builder
  - Fundamental tool for creating Cocoa user interfaces
  - Lay out UI elements
  - Specify connections
- Performance Tools
  - MallocDebug
  - ObjectAlloc
  - Shark

Recent Changes and Directions

Andreas Wendker
Senior Manager, Xcode & Core Data Frameworks

http://developer.apple.com/wwdc2005/

Recent Changes and Directions

- Many new features and enhancements in Cocoa
  - In every new release of Mac OS X
- We won’t list them all here

Topics

- Bindings
- Core Data
- Spotlight
- Error handling
- Document
- Text and Internationalization
- Archiving
- VoiceOver
- Color

Bindings and Core Data

- Two of the larger new features
- Goals
  - Significantly speed up your development process
  - Get you more functionality for less code
  - Help you build more powerful and consistent applications
  - Allow you to focus on core of your application
- Tightly integrated
- Independent of each other
  - Bindings available since Panther
  - Core Data available since Tiger

Bindings

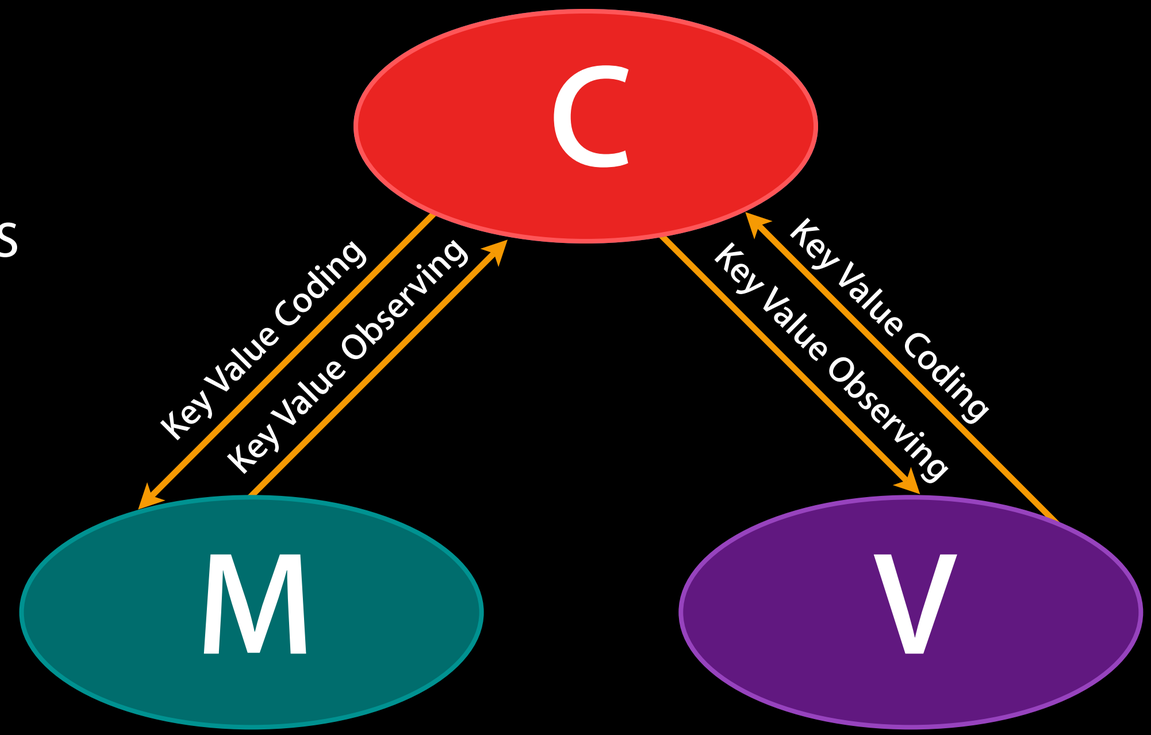
- Automate user interface synchronization
- Powerful controller classes to manage model objects
  - Sorting and filtering
  - Powerful selection handling
  - Easily display and edit object relationships
- Bindings for synchronization of model objects and views
  - Bindings usually configured code-free in Interface Builder
  - Not limited to “object values”
    - Enabled, hidden states
    - Fonts, colors
    - Table sorting, column widths
    - Tool tips

Bindings Protocols

- Based on MVC and three simple protocols
  - Key-value coding to access values by name
  - Key-value observing to observe values by name
  - Key-value binding to synchronize properties of two objects by name

```
- (void)bind:(NSString *)binding
    toObject:(id)observable
 withKeyPath:(NSString *)keyPath
     options:(NSDictionary *)options
```

- Value transformers for customizations



Bindings Adoption

- Performance: user interface refresh is well tuned
- Bindings can easily co-exist with traditional glue code
  - No need to convert entire application to benefit from Bindings
  - Mix and match in situations where bindings functionality is missing a piece you need

Core Data

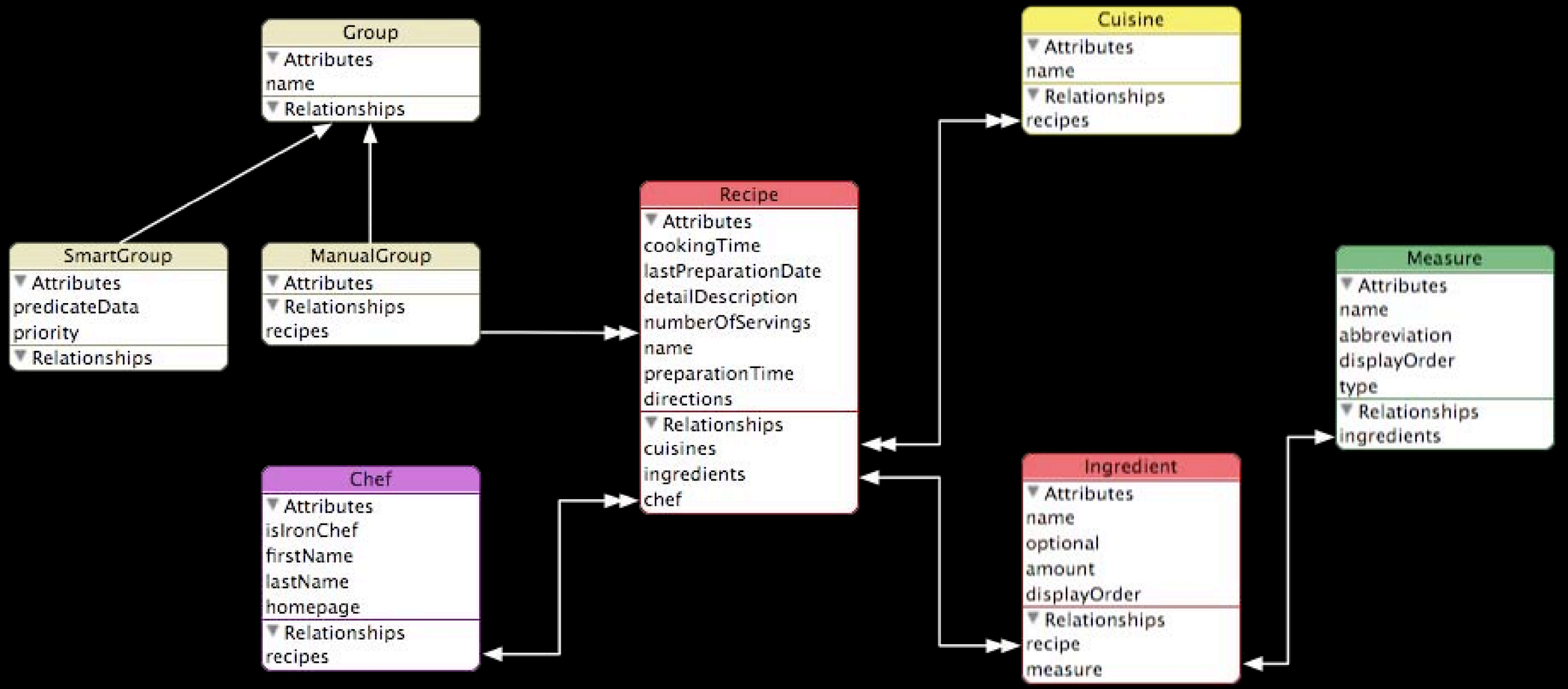
- Object oriented API to manage data models
  - Fine grained object management and persistence
- Lots of functionality for free
  - Undo/redo
  - Object value and state validation
  - Generic data objects or custom objects
- Choice of three types of object stores
  - SQLite
  - XML
  - Binary (keyed archiving)
- Object store schemas are owned by Core Data

Core Data Store Types

Store Type	Scalable to Very Large Sets of Data	Performance	Human Readable	Atomic Access
SQLite	✓	**Best**		
XML			✓	✓
Binary		**Better**		✓

Core Data Models

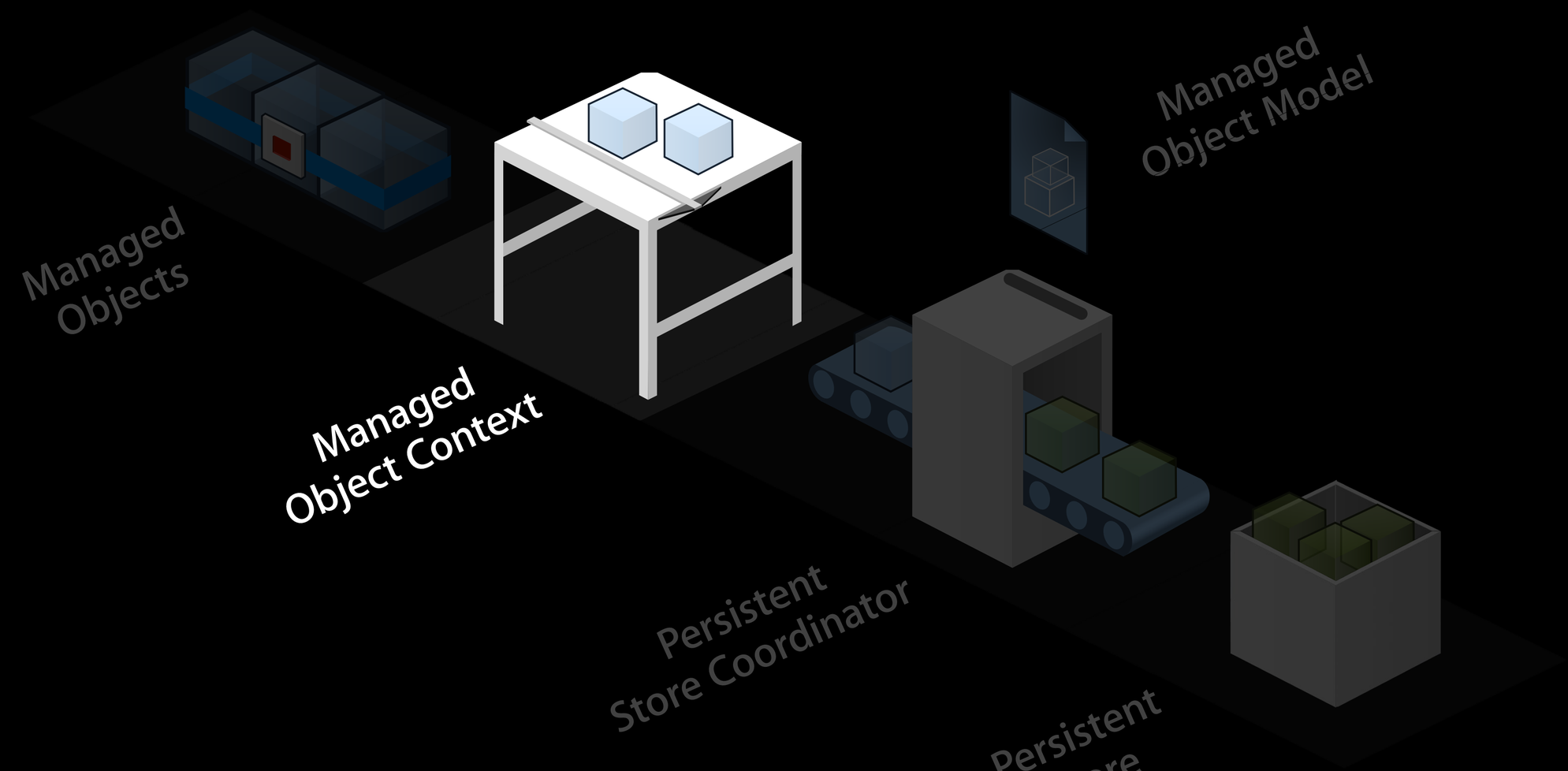
- Model-driven: create description of data objects in Xcode modeling tools
  - Entity-relationship diagram



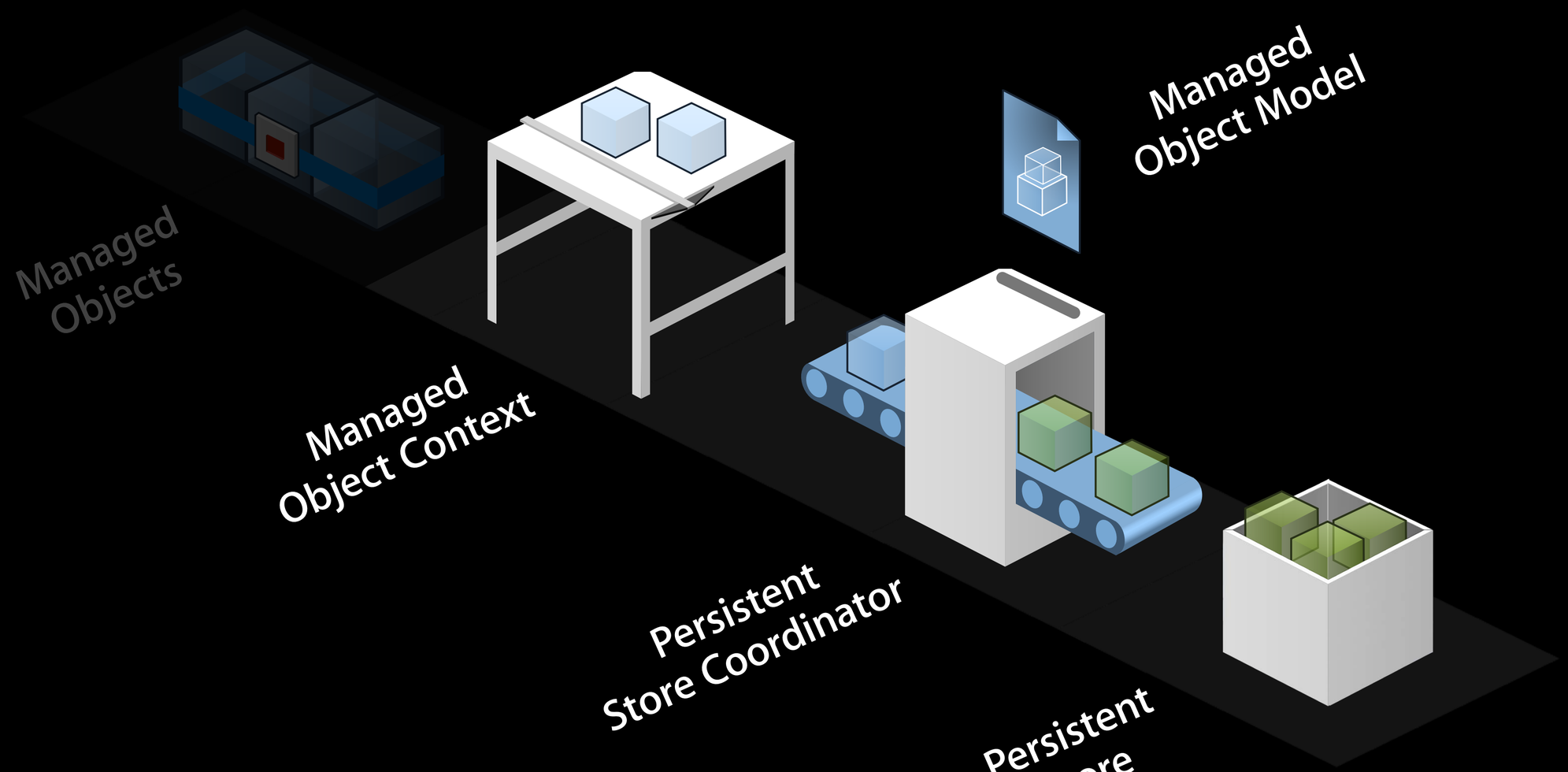
Core Data Models

- Entities contain different types of properties
  - Attributes
  - Relationships: to-one and to-many
    - Only unordered relationships supported so far (NSSets, not NSArrays)
  - Fetched Properties
- Variety of other parameters
  - Validation rules
  - Delete rules
  - Default values
  - User info dictionaries for customizations

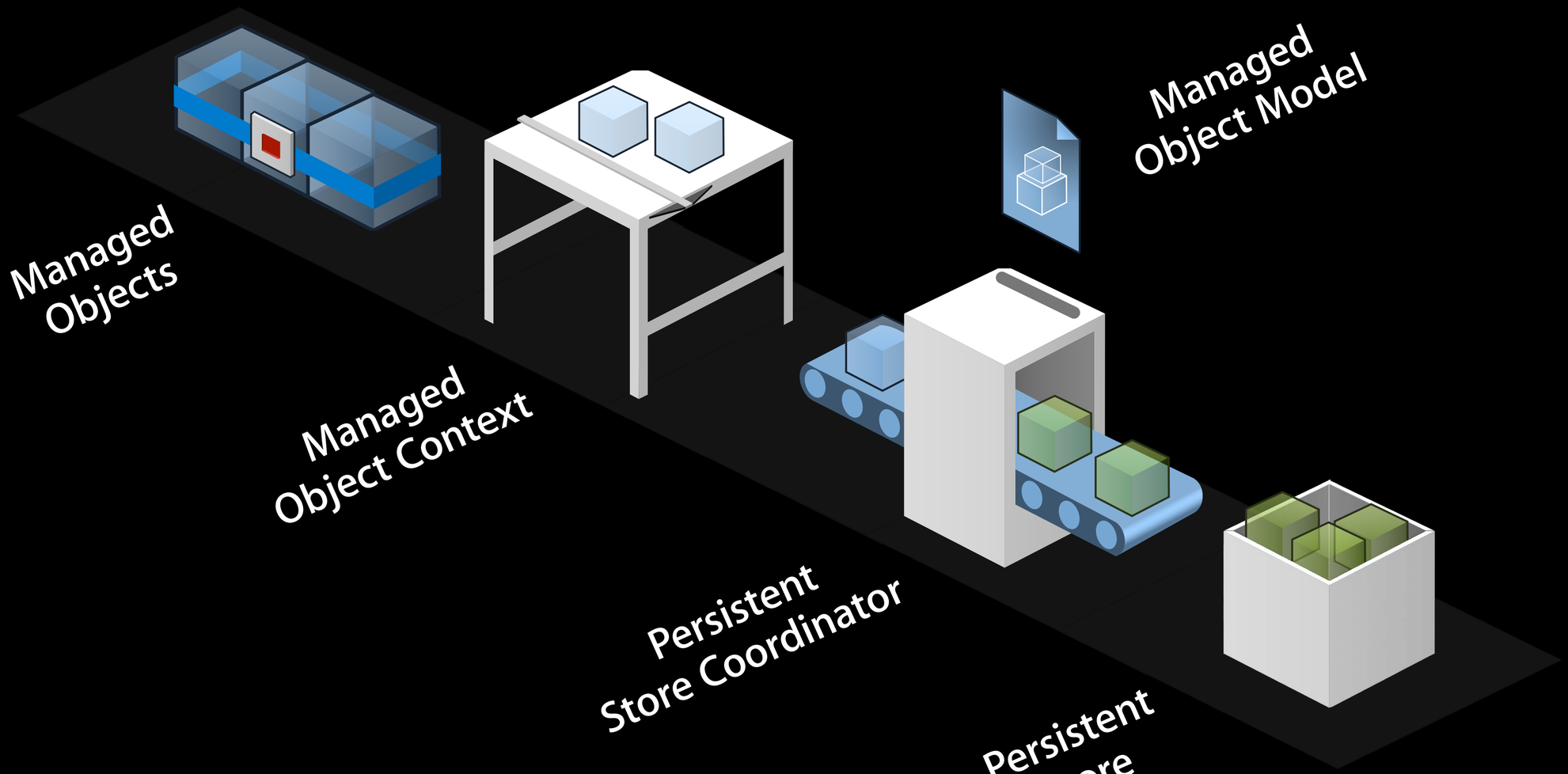
Core Data Architecture



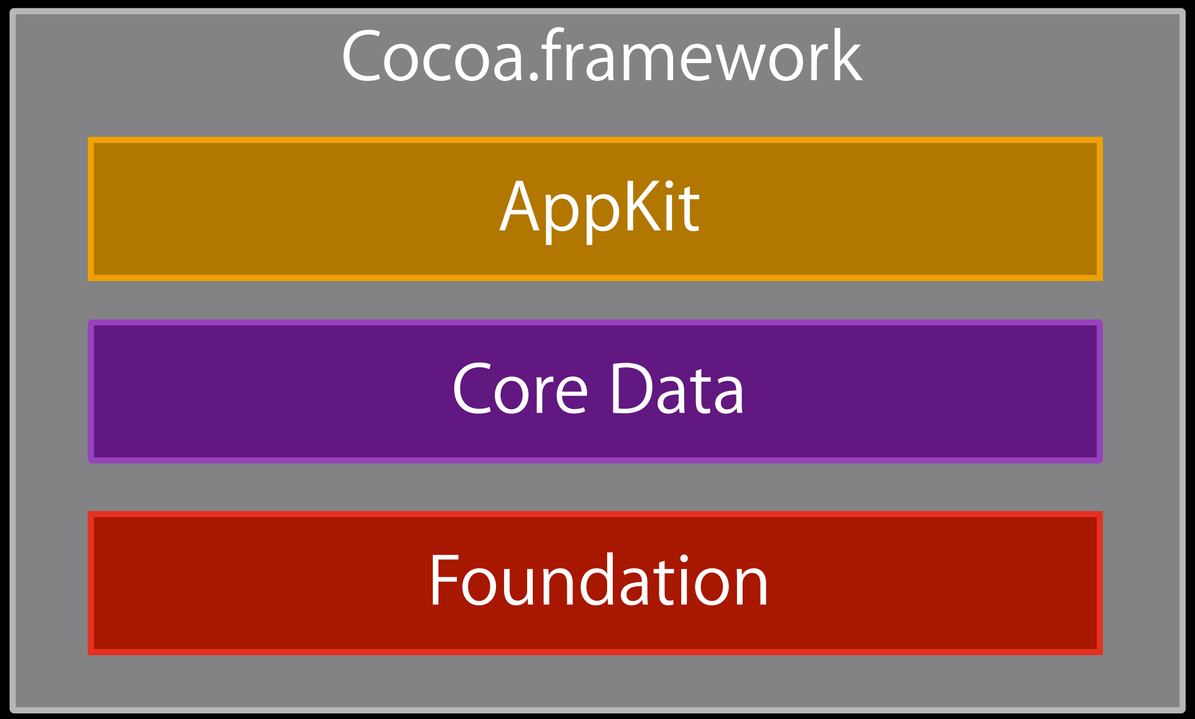
Core Data Architecture



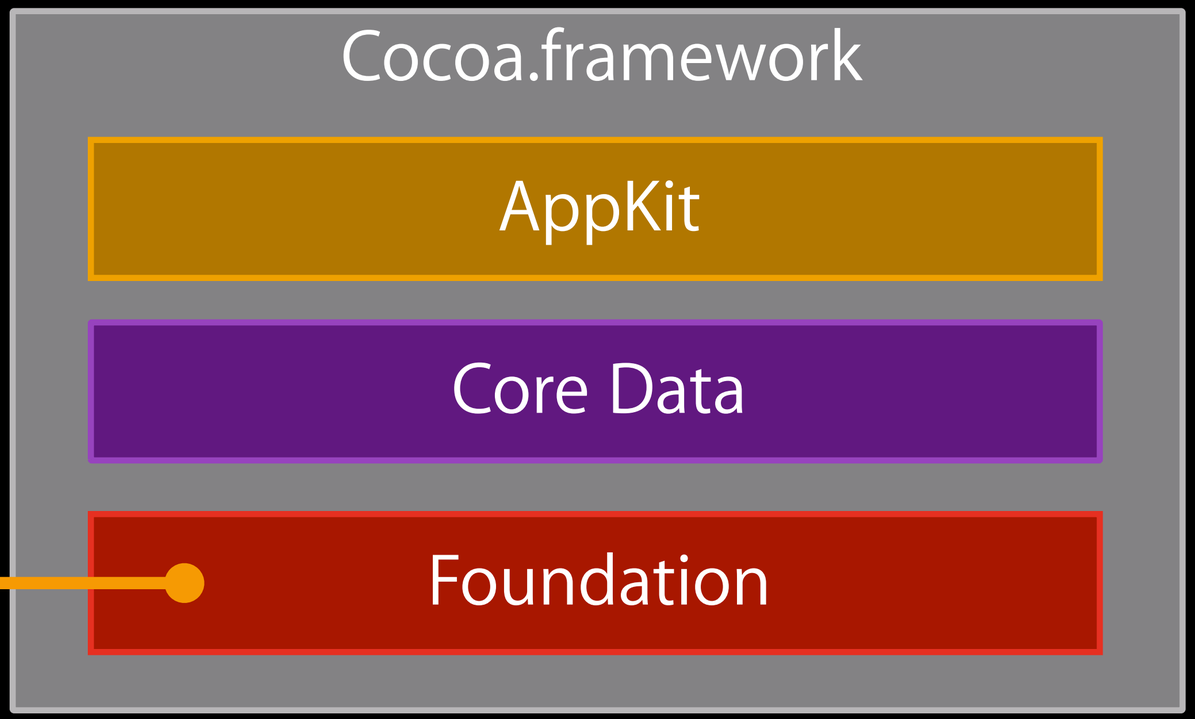
Core Data Architecture



Bindings and Core Data APIs



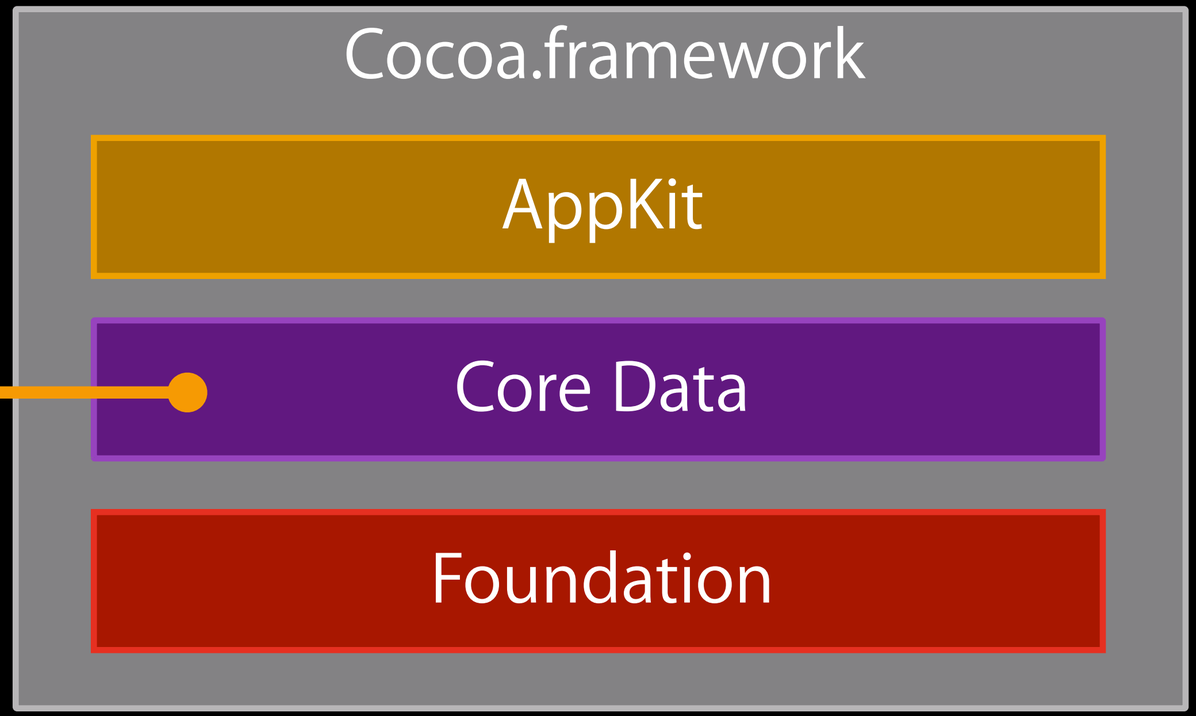
Bindings and Core Data APIs




```
NSPredicate
NSSortDescriptor
NSIndexSet
NSValueTransformer

NSKeyValueCoding protocol
NSKeyValueObserving protocol
```

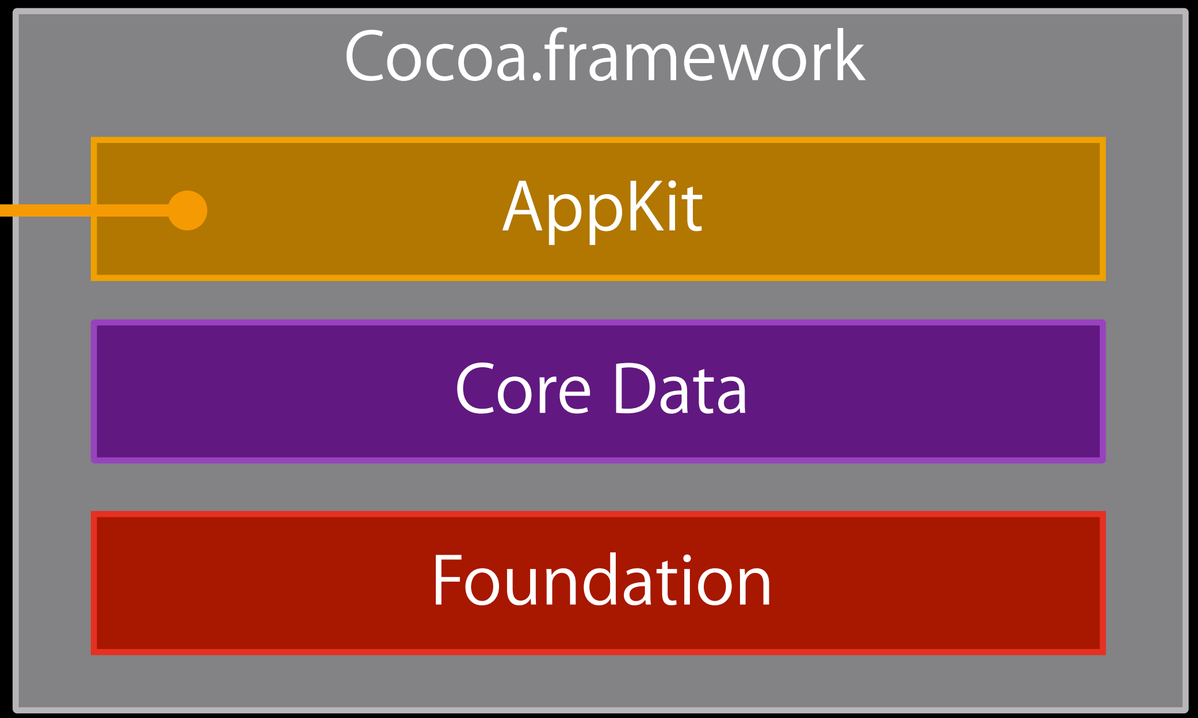
Bindings and Core Data APIs

```
NSManagedObject
NSManagedObjectID
NSManagedObjectContext
NSPersistentStoreCoordinator

NSManagedObjectModel
NSEntityDescription
NSPropertyDescription
     NSAttributeDescription
     NSFetchedPropertyDescription
     NSRelationshipDescription
```



Bindings and Core Data APIs

```
NSController
     NSObjectController
          NSArrayController
          NSTreeController
     NSUserDefaultsController

NSKeyValueBinding protocol
```



Recent Changes and Directions

Ali Ozer
Manager, Cocoa Frameworks

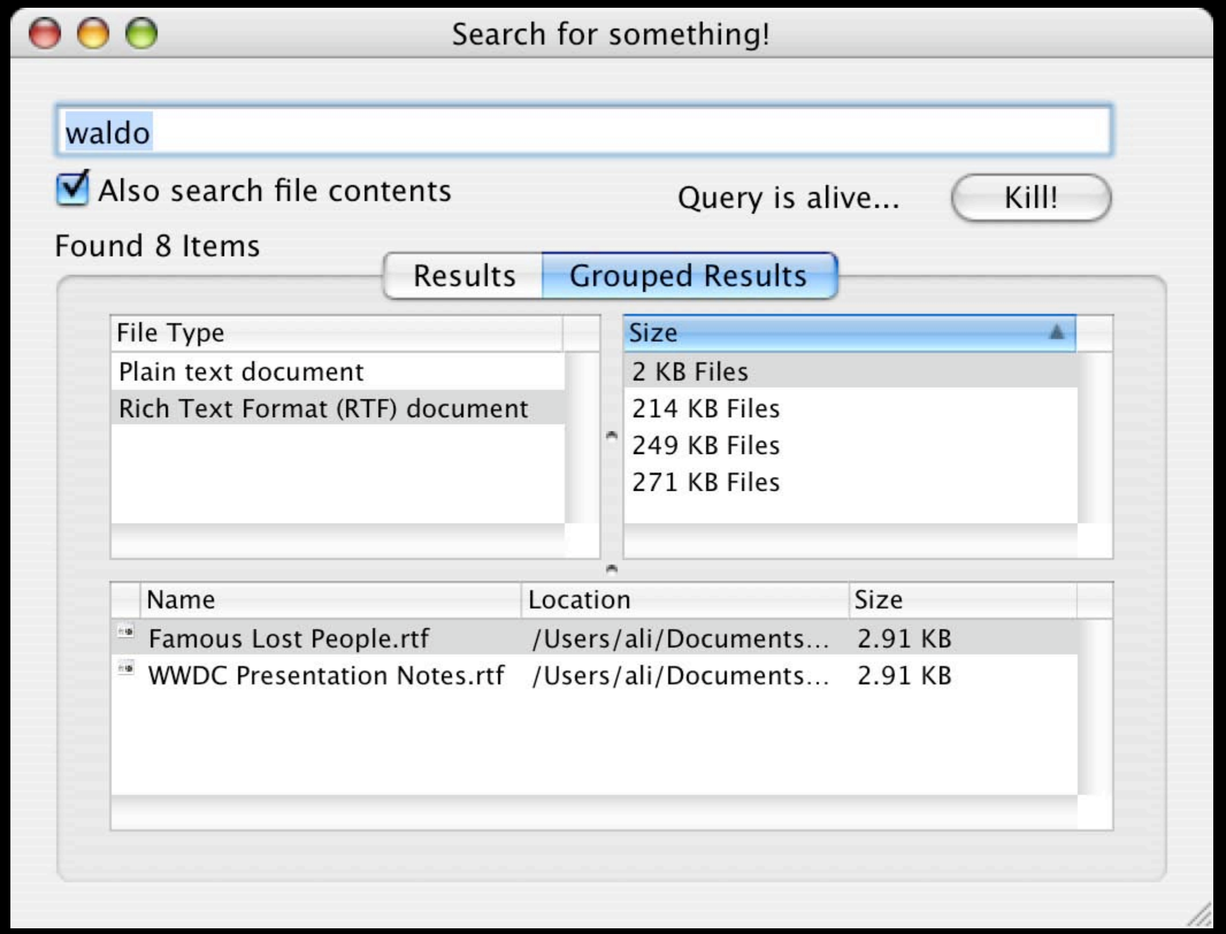
http://developer.apple.com/wwdc2005/

Spotlight

- NSMetadataQuery
  - Class for performing Spotlight queries
  - Bindable
  - Enables delegate to modify query results
- Provides
  - Query results
  - Attribute values
  - Value lists
  - Hierarchically grouped results

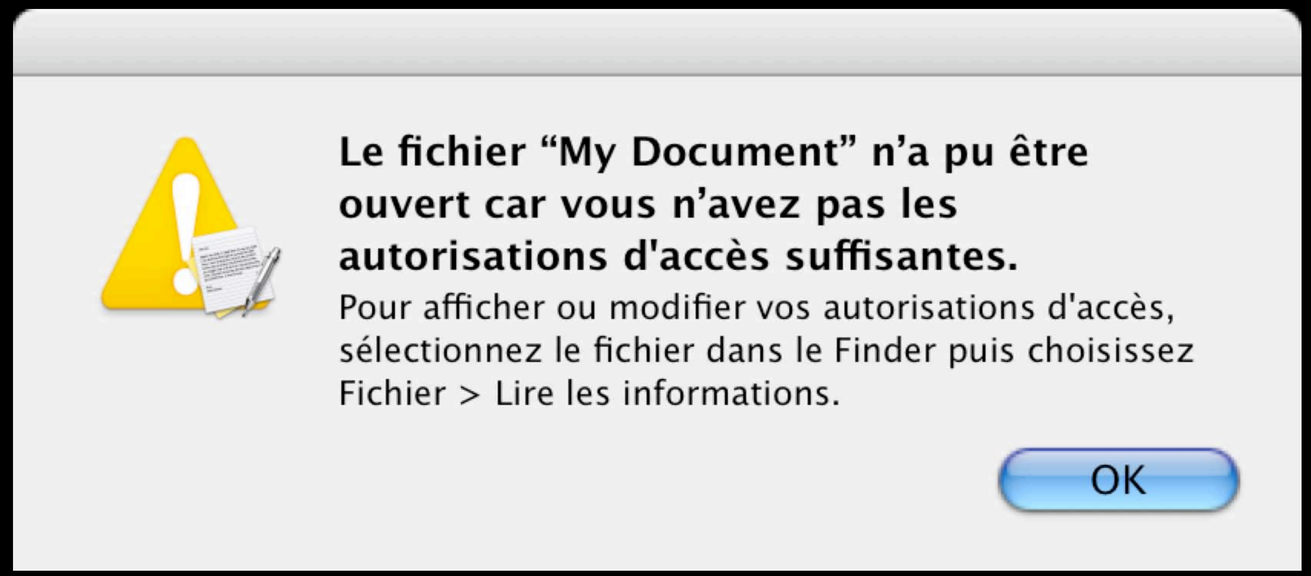
Spotlight

- /Developer/Examples/AppKit/Spotlighter



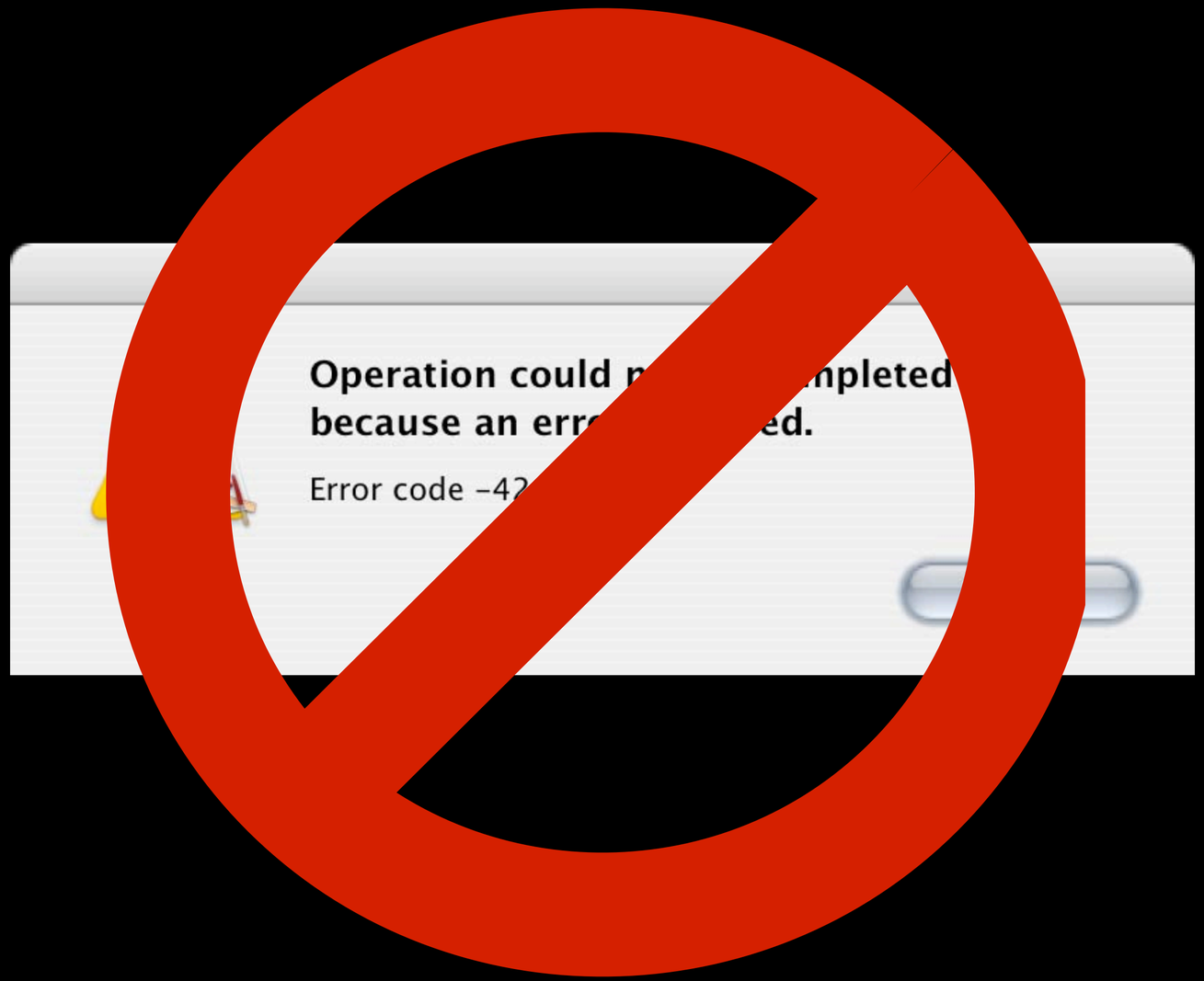
Error Handling

- NSError
  - Introduced in Panther
  - Greatly enhanced in Tiger
- Goals
  - Provide user with the polished and informative error messages
  - Handle common cases out of the box



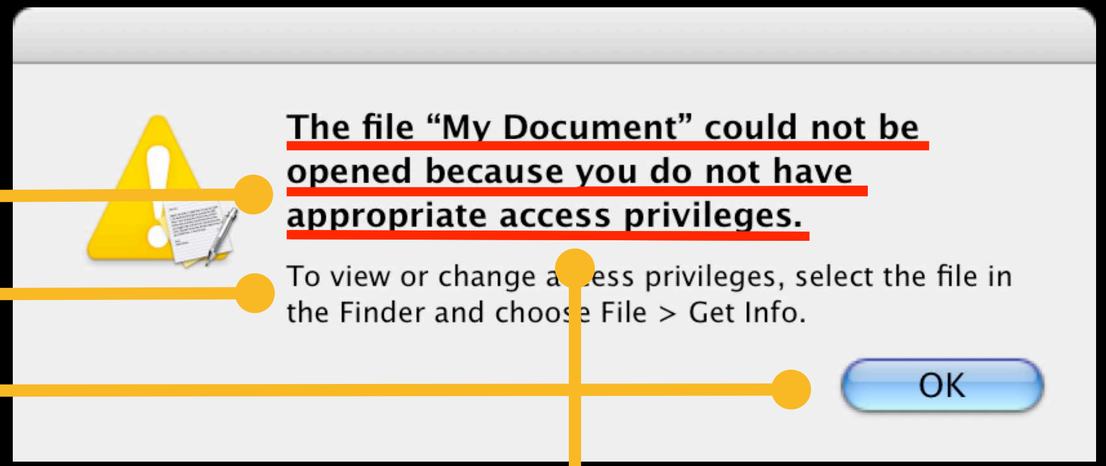
Error Handling

- Just say “NO!” to suboptimal error alerts



NSError Changes in Tiger

- New error domain
  NSCocoaErrorDomain
- New attributes
  - localizedFailureReason
  - localizedRecoverySuggestion
  - localizedRecoveryOptions
  - recoveryAttempter



localizedDescription

Error Presentation Changes in Tiger

- NSResponder-based error presentation
  - presentError:
  - presentError: modalForWindow: delegate: didPresentSelector: contextInfo:

- NSApplication delegate
  - application: willPresentError:

- NSAlert
  + alertWithError:

Classes Returning NSErrors

- NSString, NSAttributedString, NSData, NSDocument, NSXMLDocument
  - Methods to read/write files

- NSKeyValueCoding

```
- (BOOL) validateValue: (id *)ioValue
                forKey: (NSString *)inKey
                 error: (NSError **) errorPtr;
```

- NSNumberFormatter, NSDateFormatter

```
- (BOOL) getObjectValue: (id *)obj
              forString: (NSString *)string
                  range: (NSRange *)rangePtr
                  error: (NSError **)errorPtr;
```

NSDocument

- Better error handling
- NSPersistentDocument
- Autosaving

NSDocument Autosaving

- Periodic autosaving of open documents
- Just takes one call to enable it all:
  - setAutosavingDelay:
- A number of other related methods, in case you need them

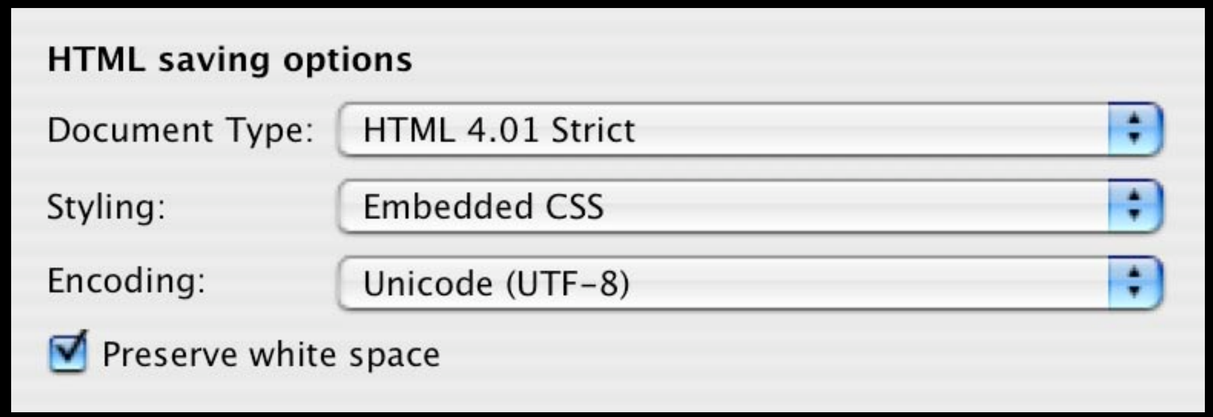
NSText

- Document formats
- Document properties

Text Document Formats

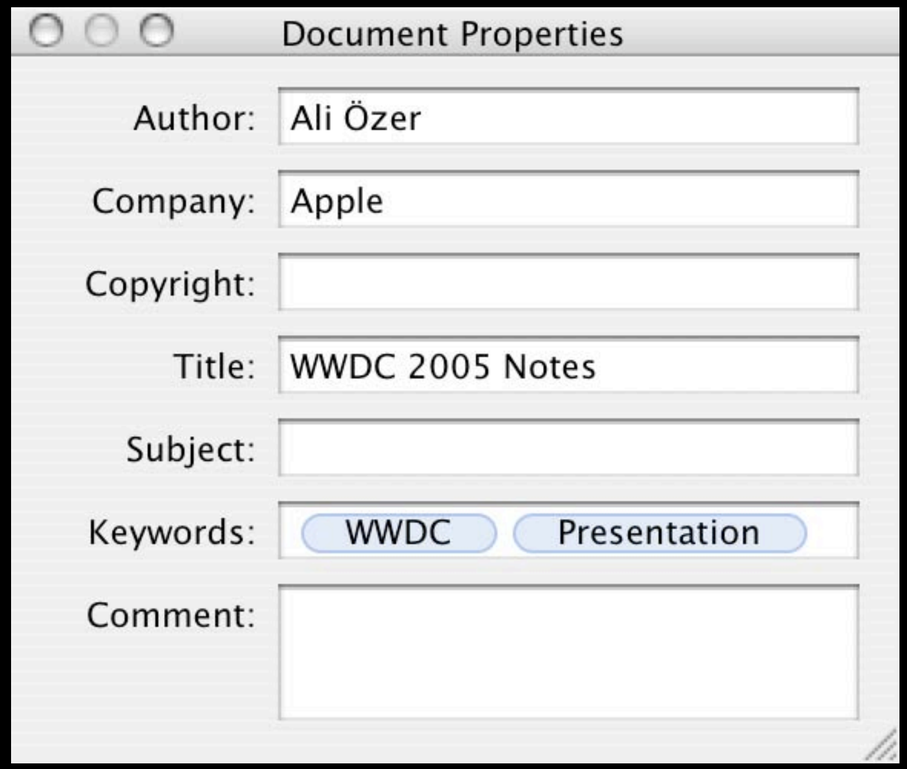
- Text system now reads plain text, RTF, RTFD, HTML, WebArchive, Word, WordML, and SimpleText
  - Writes all but SimpleText
- Document attributes are available to control the type of HTML generated

  NSExcludedElementsDocumentAttribute
  NSPrefixSpacesDocumentAttribute



Text Document Properties

- Read/write document metadata in text documents
  - NSTitleDocumentAttribute
  - NSCompanyDocumentAttribute
  - NSSubjectDocumentAttribute
  - NSAuthorDocumentAttribute
  - NSKeywordsDocumentAttribute
  - NSCommentDocumentAttribute
  - NSEditorDocumentAttribute
  - NSCreationTimeDocumentAttribute
  - NSCopyrightDocumentAttribute



Other Changes in the Text System

- Text tables
- Text lists
- Multiple selection
- Standard panels to edit links, spacing, lists, tables
- Improved text layout for complex scripts

Internationalization

- New classes
  - NSLocale
  - NSCalendar

- Updated classes
  - NSDateFormatter
  - NSNumberFormatter
  - NSString

NSLocale

- Key/value based API to extract locale info
  - Refer to documentation or NSLocale.h for the keys
- Basis for future improvements
- Existing locale: arguments still expect NSDictionary instances

NSCalendar

- Compose/decompose date components
- Perform calendar arithmetic
- Query numeric properties
- Refer to documentation or NSLocale.h for list of predefined instances

```
NSRange daysRange = [cal rangeOfUnit:NSDayCalendarUnit
                              inUnit:NSMonthCalendarUnit
                             forDate:[NSDate date]];

NSDateComponents *comps = [cal components:NSDayCalendarUnit
                                 fromDate:[NSDate date]
                                   toDate:christmas
                                  options:0];
```

NSDateFormatter

- Major implementation upgrade
  - Based on ICU
  - New specifiers in the format string
- New formatter features enabled by selecting “10.4” behavior

```
[formatter setFormatterBehavior:NSDateFormatterBehavior10_4];
```

- Specify “style” of date and time desired
  - NoStyle
  - ShortStyle 6/7/05 14:42
  - MediumStyle Jun 7, 2005 14:42:30
  - LongStyle June 7, 2005 14:42:30 PDT
  - FullStyle Tuesday, June 7, 2005 14:42:30 PDT

NSNumberFormatter

- Same situation as NSDateFormatter
  - Based on ICU
  - Some changes in the format string specifiers
- Specify "style" of number desired
  - NoStyle
  - DecimalStyle 3.142
  - CurrencyStyle $3.14
  - PercentStyle 314%
  - ScientificStyle 3.14159265E0
  - SpellOutStyle three point one four one five nine two six five

NSDateFormatter, NSNumberFormatter

- More easily usable in contexts outside of cells
  - NSDateFormatter

```
- (NSString *) stringFromDate: (NSDate *)date;
- (NSDate *) dateFromString: (NSString *)string;
```

  - NSNumberFormatter

```
- (NSString *) stringFromNumber: (NSNumber *)number;
- (NSNumber *) numberFromString: (NSString *)string;
```

NSString

- Get off of “C-String” APIs

Get Off of "C-String" APIs

- NSString APIs which do not specify an encoding explicitly
  - cString
  - getCString:
  - initWithCString:
  - initWithContentsOfFile:
  - initWithContentsOfURL:

    etc.
- Full list declared in the NSStringDeprecated protocol in NSString.h

Get Off of "C-String" APIs

- Instead use Panther or Tiger introduced methods such as
  - cStringUsingEncoding:
  - getCString: maxLength: encoding:
  - initWithCString: encoding:
  - initWithContentsOfFile: encoding: error:
  - initWithContentsOfURL: encoding: error:
  - initWithContentsOfFile: usedEncoding: error:
  - initWithContentsOfURL: usedEncoding: error:

Keyed Archiving

- Enables creating archives which are more robust and flexible
- Values are tagged with keys
  - Instead of

```
[archiver encodeObject: obj];
obj = [unarchiver decodeObject];
```

  - with keyed archiving you do:

```
[archiver encodeObject: obj forKey: @"title"];
obj = [unarchiver decodeObjectForKey: @"title"];
```

- Works back to Mac OS X Jaguar (10.2)

VoiceOver

- Apple’s built-in accessibility interface
  - Magnification options
  - Keyboard control
  - Spoken descriptions
- Ready to go right out of the box
  - Enabled during installation
  - Or hit Command-F5 to enable on the fly
- Makes use of accessibility APIs added to Mac OS X in Jaguar

Accessibility

- Now features ability to set attribute values for any accessible UI element
  - accessibilitySetOverrideValue: forAttribute:
- New attributes
  - Enable spoken descriptions for UI elements without text

    NSAccessibilityDescriptionAttribute
  - Link UI elements together

    NSAccessibilityLinkedUIElementsAttribute
- Interface Builder now provides ability to edit attributes

Color Space Support in NSColor

- Methods such as colorWithCalibratedRed:green:blue:alpha: now do return “calibrated” colors
  - Corresponding to “generic” color spaces in Quartz

Color Space Support in NSColor

- New class NSColorSpace enables custom color space support
  - Predefined instances

```
+ (NSColorSpace *) genericRGBColorSpace;
```

  - Instances created from ICC or ColorSync profile data

```
- (id) initWithICCProfileData: (NSData *)iccData;
- (id) initWithColorSyncProfile: (CMProfileRef)prof;
```

  - To create NSColor with a custom color space

```
+ (NSColor *) colorWithColorSpace: (NSColorSpace *)space
                       components: (const float *)components
                            count: (int)numberOfComponents;
```

Color Space Support in NSColor

- Color panel enables users to select other color spaces

Stuff You Should Not Be Doing

- Assuming sizeof(int) == sizeof(void *)
  - 64-bit
- Assuming 1 pixel == 1 point
  - Resolution independence
- Assuming APIs return things other than what they say
  - NSUserDefaults change
- Drawing faster than 60 fps
  - Coalesced updates
- Using private APIs

Universal Considerations

- File and network data byte ordering issues
  - Examples
    - UTF-16 files
    - Files with ints, structs, etc.
    - Archiving bit fields
  - Possible solutions
    - XML files
    - NSKeyedArchiver
    - NSByteOrder functions
    - Appropriate high-level APIs to read/write file formats

Universal Considerations

- gcc 4.0
- Direct use of OpenGL, CGBitmapContexts, CGImageRefs
- Bugs/undefined behaviors
  - Uninitialized variables
  - Invalid memory accesses
  - Integer shift left by 32
  - Integer divide or mod by 0
- Sending Objective-C messages with floating point or struct returns to nil

Mike Rossetti

QuickBooks Engineering, Intuit Software

http://developer.apple.com/wwdc2005/

Cocoa: It's Intuit'ive

Brought to you by: The Intuit QuickBooks Team

http://developer.apple.com/wwdc2005/

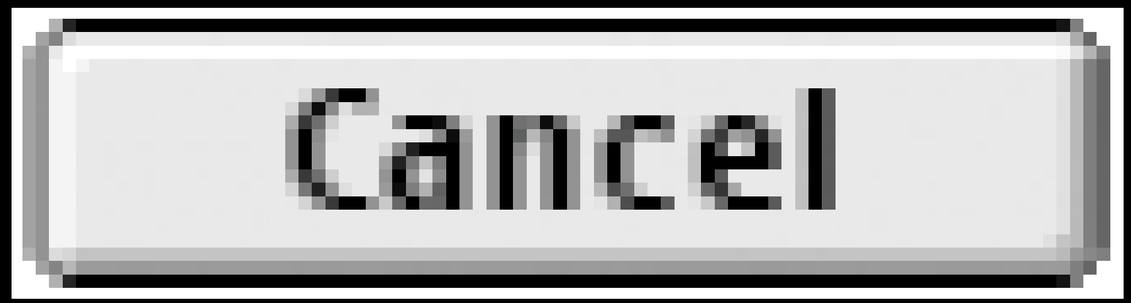
Archeology

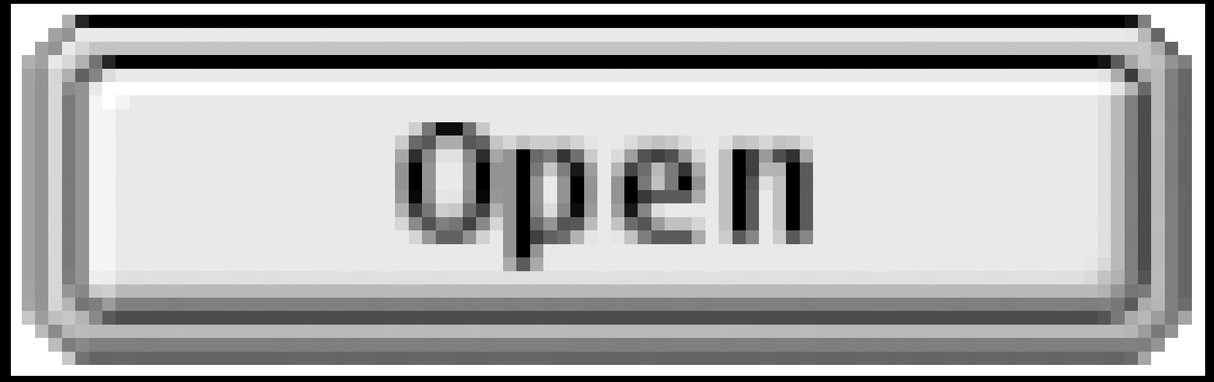
- A bit over three years ago
- Quite small team
- Targets: Mac OS 9 and Mac OS X
- “You can’t start from scratch!”
- Pretty ancient C code base

Hysterical Markers

- “01/01/85 AJM Initial Revision”

What's the PtInRect?

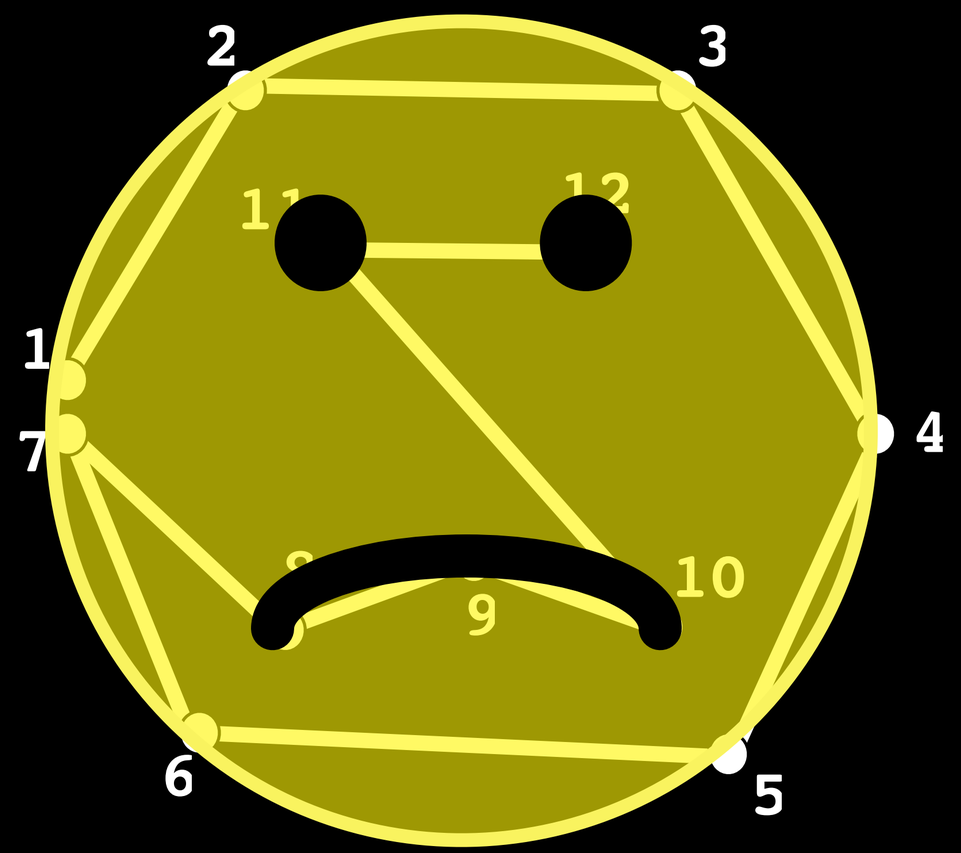




Control Manager

C++ Wannabe

- Switch/Case..Case..Case..Case..Case..Case..
- Deep callbacks



Year 1: Into the Carbon Age...

- Build-able (CodeWarrior)
- Carbonized (some PP)
- Aqua-fied UI
- Core Foundation
- Groomed code

Year 2: Momentous Events

- Moved to pure Carbon events
- Added new features
- Used C++ and Standard Library

Year 3: Hot Cocoa

- Carbon-in-Cocoa
- Mostly Carbon windows, some Cocoa
- Cocoa menu management
- Bridged Carbon/PP/Cocoa events
- .Mac support

Year 4: It was Bind to Happen

- Payback begins big-time
- New features in pure Cocoa
- Cocoa Bindings
- SQLite3
- WebKit integration
- Sync Services

> Core Data

The Moral?

- You don’t have to start from scratch
- Obj-C and C++ play nicely together
- This stuff is FUN
- The power of frameworks is incredibly important
- **The benefits are tremendous**

COCOA IS CLOSER THAN IT
MAY APPEAR.

More Information

http://developer.apple.com/wwdc2005/

Documentation, Sample Code, and Other Resources

http://developer.apple.com/wwdc2005/

Who to Contact

Matthew Formica
Cocoa and Developer Tools Evangelist
mformica@apple.com

Q&A

Matthew Formica
Cocoa and Developer Tools Evangelist
mformica@apple.com

http://developer.apple.com/wwdc2005/

WWDC2005

Worldwide Developers Conference